

(19)日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11)特許出願公開番号

特開平11-134227

(43)公開日 平成11年(1999) 5月21日

(51)Int.Cl.⁸

G 0 6 F 12/00

識別記号

5 1 1

F I

G 0 6 F 12/00

5 1 1

審査請求 未請求 請求項の数28 O L (全 21 頁)

(21)出願番号 特願平9-293765

(22)出願日 平成9年(1997)10月27日

(71)出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72)発明者 猪原 茂和

東京都国分寺市東恋ヶ窪一丁目280番地

株式会社日立製作所中央研究所内

(72)発明者 鍵政 豊彦

東京都国分寺市東恋ヶ窪一丁目280番地

株式会社日立製作所中央研究所内

(72)発明者 野田 文雄

東京都国分寺市東恋ヶ窪一丁目280番地

株式会社日立製作所中央研究所内

(74)代理人 弁理士 小川 勝男

最終頁に続く

(54)【発明の名称】 ファイルフォーマット変換方法及びこれを用いたファイルシステム及び情報システム及び電子商取引システム

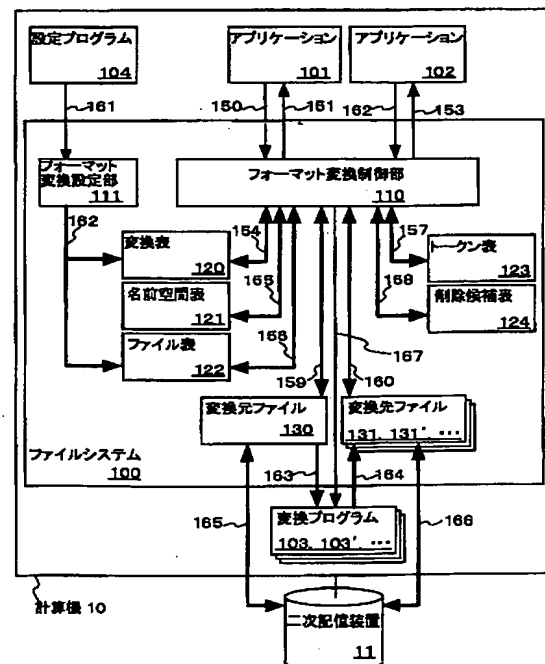
(57)【要約】

【課題】 複数のファイルフォーマットのファイルの間のフォーマット変換をユーザの関与なしに行う。

【解決手段】 ファイルシステム100が変換元ファイル130と変換先ファイル131の関係を保持し、ファイル操作APIの発行に同期してフォーマット変換に関する処理を行う。

【効果】 ユーザはアプリケーション上で行うユーザの本来の作業を行えばよくなり、その過程で必要となる様々なフォーマット変換(1段階でも、多段階でも良い)はユーザの関与なしで行われる。この際、変換元ファイルを指定したり、フォーマット変換のタイミングを指定する必要がなく、ユーザは常に最新の変換先ファイルを利用可能となる。

図1



・【特許請求の範囲】

【請求項1】 所定のフォーマットを有する第1のファイルを変換プログラムにより該フォーマットと異なるフォーマットを有する第2のファイルに変換するファイルフォーマット変換方法であって、(1) ユーザのアプリケーションにより第2のファイルの1つを指定し、(2) 第1のファイルと、該第1のファイルから該指定した第2のファイルを得る変換プログラムとを対応づけ、

(3) 該変換プログラムにより第1のファイルから少なくとも1つの所望の第2のファイルへのフォーマット変換を行う、ファイルフォーマット変換方法。

【請求項2】 前記(1)の指定する処理は、前記第2のファイルの1つに対してユーザのアプリケーションが発行するファイル操作を用いてなされる請求項1記載のファイルフォーマット変換方法。

【請求項3】 第1のフォーマットを有する第1のファイルを変換プログラム P_1, P_2, \dots, P_m ($m: 1$ 以上の整数)によりそれぞれ第1のフォーマットと異なる第2のフォーマットを有する第2のファイル F_1, F_2, \dots, F_m に変換するファイルフォーマット変換方法であって、

(1) 第1のファイルと変換プログラムとそれぞれの変換プログラムにより変換される第2のファイルとを対応づけ、(2) ユーザのアプリケーションにより第1のファイル又は第2のファイルを指定し、(3) ①ユーザのアプリケーションが発行する第1のファイル又は第2のファイルへのファイル操作の時刻、②所定の時刻、③所定の条件が成立した時刻、のうちの少なくとも1つの時刻を契機として、上記対応づけられた変換プログラムにより第1のファイルから第2のファイルへのフォーマット変換を行う、ファイルフォーマット変換方法。

【請求項4】 第1のフォーマットを有する第1のファイルを変換プログラム P_1, P_2, \dots, P_m ($m: 1$ 以上の整数)によりそれぞれ第1のフォーマットと異なる第2のフォーマットを有する第2のファイル F_1, F_2, \dots, F_m に変換するファイルフォーマット変換方法であって、

(1) 第1のファイルと変換プログラムとそれぞれの変換プログラムにより変換される第2のファイルとを対応づけ、(2) 上記変換プログラムはそれぞれ上記第1のファイルから第3のファイルを得る第1の中間変換プログラムと、第3のファイルから上記第2のファイルを得る第2の中間変換プログラムとからなり、(3) ユーザのアプリケーションにより第1のファイル又は第2のファイルを指定し、(4) アプリケーションが発行する第1のファイルへのファイル操作を第1の契機として上記対応する変換プログラムの第1の中間変換プログラムにより第1のファイルから第3のファイルを得、該第3のファイルへのファイル操作を第2の契機として第2の中間変換プログラムにより第3のファイルから第1のファイルを得る、ファイルフォーマット変換方法。

【請求項5】 前記第1の契機として、第1のファイルへ

の書き込み操作または書き込みのためのオープン操作を用い、前記第2の契機として第2のファイルへの読み出し操作または読み出しのためのクローズ操作を用いる請求項4記載のフォーマット変換方法。

【請求項6】 前記第1のファイルが第1の計算機に格納され、前記第2のファイルと前記第3のファイルが第2の計算機に格納される請求項4記載のフォーマット変換方法。

【請求項7】 第1のフォーマットを有する第1のファイルを変換プログラム P_1, P_2, \dots, P_m ($m: 1$ 以上の整数)によりそれぞれ第1のフォーマットと異なる第2のフォーマットを有する第2のファイル F_1, F_2, \dots, F_m に変換するファイルフォーマット変換方法であって、

(1) 第1のファイルと変換プログラムとそれぞれの変換プログラムにより変換される第2のファイルとを対応づけ、(2) 第1のファイルのファイル名から第2のファイルのファイル名を生成し、(3) 生成したファイル名で第2のファイルをユーザに提供し、(4) 提供されたユーザのアプリケーションにより該ファイル名で第1のファイル又は第2のファイルを指定し、(5) アプリケーションが発行する第1のファイル又は第2のファイルへのファイル操作を契機として上記対応づけられた変換プログラムにより第1のファイルから第2のファイルへのフォーマット変換を行う、ファイルフォーマット変換方法。

【請求項8】 前記(2)のファイル名を生成する処理は、前記第1のファイルの作成操作を契機として行う請求項7記載のファイルフォーマット変換方法。

【請求項9】 前記(2)のファイル名を生成する処理は、第2のファイルのファイル名を含むディレクトリに対する検索操作または表示操作を契機として行う請求項7記載のファイルフォーマット変換方法。

【請求項10】 前記(1)の対応づける処理は、第1のファイルと、第1の変換プログラムのうちのひとつと、当該変換プログラムにより第1のファイルから得られる第2のファイルの3種のうち、第1のファイルから対応する2種を得る方法、または、第2のファイルから対応する2種を得る方法を含む請求項3乃至9いずれか1項記載のファイルフォーマット変換方法。

【請求項11】 前記対応づける方法として、第1のファイルと、第1の変換プログラムと、第2のファイルとの間の対応関係を保持する表を用いる請求項10記載のファイルフォーマット変換方法。

【請求項12】 前記対応づける方法として、第1のファイルと、第1の変換プログラムと、第2のファイルとの間の対応関係を保持するプログラムを用いる請求項10記載のファイルフォーマット変換方法。

【請求項13】 前記ファイル操作は、第1のファイルへの書き込み操作か書き込み後のクローズ操作、または第2のファイルへの読み出し・書き込み操作か読み出し・

書き込みのためのオープン操作である、請求項3、4、6、7いずれか1項記載のファイルフォーマット変換方法。

【請求項14】前記第1及び第2のファイルと異なる第3のファイルがあり、前記第1のファイルから第2のファイルへのフォーマット変換時に、前記第3のファイルから第1のファイルへのフォーマット変換を行う請求項1乃至13項いずれか1項記載のファイルフォーマット変換方法。

【請求項15】前記第1のファイルまたは前記第2のファイルのいずれかが、電子メール、または電子メールの添付ファイルである請求項1乃至14いずれか1項記載のファイルフォーマット変換方法。

【請求項16】第1のファイルと、第1のファイルをフォーマット変換して得られる1つ以上の第2のファイルをユーザに提供するファイルフォーマット変換方法であって、第1のファイルに対する第1のファイル操作と、第2のファイルの1つである第3のファイルに対する第2のファイル操作のうち、同時に実行する操作をただか1つとする、ファイルフォーマット変換方法。

【請求項17】前記第1のファイル操作と前記第2のファイル操作のどちらかが、書き込み操作、または書き込みモードのオープン操作である請求項16記載のファイルフォーマット変換方法。

【請求項18】フォーマット変換によって得た前記第2のファイルの一部または全部について、ファイル名を削除せずにファイルの内容を削除する請求項1乃至17いずれか1項記載のファイルフォーマット変換方法。

【請求項19】請求項1ー18いずれか1項記載のファイルフォーマット変換方法をライブラリとして記録した記録媒体。

【請求項20】複数のファイルを保持する二次記憶装置を具備した計算機システムに搭載されるファイルシステムであって、変換元ファイルと、フォーマット変換後の変換先ファイルと、変換元ファイルと変換プログラムと変換先ファイルとを対応づける対応指示部と、アプリケーションプログラムが行い得るファイル操作を規定するアプリケーション・プログラミング・インターフェース（以下、API）と、該APIの起動に応じて所望のフォーマット変換を実行するフォーマット変換制御部とからなる、ファイルシステム。

【請求項21】複数のファイルを保持する二次記憶装置を具備した計算機システムが複数個ネットワークを介して接続されている分散システムにおける各計算機システムに搭載されるファイルシステムであって、変換元ファイルとフォーマット変換後の変換先ファイルとの少なくとも一方からなるファイルと、変換元ファイルと変換プログラムと変換先ファイルとを対応づける対応指示部と、アプリケーションプログラムが行い得るフ

ァイル操作を規定するアプリケーション・プログラミング・インターフェース（以下、API）と、ネットワークを介して他のファイルシステムと相互に通信を行いながら該APIの起動に応じて所望のフォーマット変換を実行するフォーマット変換制御部とからなる、ファイルシステム。

【請求項22】前記対応指示部は、対応づけを登録または削除するインターフェースを含む請求項20または21項記載のファイルシステム。

【請求項23】ファイル編集の機能を有し電源のオン／オフが起こりうる第1のパーソナルコンピュータ（以下、PC）と、電源がオン状態になっているWorld Wide Web（以下、WWW）サーバ計算機と、ファイル変換の機能を有しない第2のパーソナルコンピュータ（以下、TPC）とをネットワークで結合した情報システムであって、

PCは、所定のフォーマットを有する変換元ファイルと、アプリケーションプログラムが行いうるファイル操作を規定するアプリケーション・プログラミング・インターフェース（以下、API）を通したユーザからの指示に基づき内蔵の変換プログラムによるフォーマット変換を制御する制御部とを備え、

WWWサーバ計算機は、PCにおけるフォーマット変換で得られた中間ファイルと、該中間ファイルを少なくとも1個の変換プログラムによりフォーマット変換して得られた少なくとも1個の変換先ファイルと、APIを通したユーザからの指示に基づき該フォーマット変換を制御するフォーマット変換制御部とを備え、

TPCは、WWWサーバ計算機中にある上記変換先ファイルの少なくとも1個をWWWブラウザ経由で指定してその内容の読み出しをWWWサーバ計算機に対して指示する機能を有する、情報システム。

【請求項24】複数のファイルを保持する二次記憶装置を備えたパーソナル・コンピュータであって、該パーソナル・コンピュータは、ファイル編集の機能を備えるアプリケーションと、1つ以上のフォーマットのうちののあるフォーマットを有する文書の表示プログラムと、変換元ファイルを変換先ファイルにフォーマット変換するために請求項1ー7いずれか1項に記載のファイルフォーマット変換方法を有するファイルシステムとを備え、前記アプリケーションは、ユーザの操作に従って変換元ファイルを作成して二次記憶装置に保存し、表示プログラムは、オープンAPIを発行し、該発行に同期して前記ファイルシステムにおいて変換元ファイルから変換された他の変換先ファイルの内容を読み出して表示する、パーソナル・コンピュータ。

【請求項25】複数のファイルを保持する二次記憶装置を備えたサーバ計算機であって、該サーバ計算機は、ファイル編集機能を有するアプリケーションと、1つ以上のフォーマットのうちののあるフォーマットを有する文書

を表示または編集するプログラムと、変換元ファイルを変換先ファイルにフォーマット変換するために請求項1ー15いずれか1項記載のファイルフォーマット変換方法を有するファイルシステムとを備え、前記アプリケーションは、ユーザの操作に従って変換元ファイルを作成して二次記憶装置に保存し、WWWサーバは、機能限定パーソナル・コンピュータ（以下、PCという）とネットワークで結合され、該機能限定PCに搭載された前記プログラムからのファイル読み出し要求を受けてオープンAPI（アプリケーション・プログラミング・インターフェース）を発行し、該発行に同期して前記ファイルシステムにおいて前記変換元ファイルから変換された変換先ファイルの内容を読み出して前記プログラムに返送し、分散ファイル・サーバは、変換先ファイルフォーマットの1つを有する文書を表示または編集するプログラムに前記ネットワーク経由で結合され、該プログラムから変換先ファイルの読み出し要求を受けてオープンAPIを発行し、該発行に同期して前記ファイルシステムにおいて変換元ファイルから変換された変換先ファイルの内容を読み出して該プログラムに返送する、サーバ計算機。

【請求項26】複数のファイルを保持する二次記憶装置を備えた検索サーバ計算機であって、WWWクライアントと、少なくとも1つの検索サーバと、変換元ファイルを変換先ファイルにフォーマット変換するために請求項1ー15いずれか1項記載のファイルフォーマット変換方法を有するファイルシステムとを備え、WWWクライアントは、それぞれ異なるフォーマットを有するファイルを保持した複数のWWWサーバ計算機とネットワークで結合され、各ファイルの内容を収集してそれぞれ変換元ファイルとして前記ファイルシステムに格納し、検索サーバは、検索すべき情報の種類対応に設けられ、前記ファイルシステムにおいて異なるフォーマットを有する変換元ファイルをフォーマット変換した後の同一フォーマットを有する変換先ファイル群に対して前記情報の種類に対応した検索サーバによる検索を行う、検索サーバ計算機。

【請求項27】前記検索サーバは、第1のフォーマットを入力として受け取る第1の検索プログラムと第2のフォーマットを入力として受け取る第2の検索プログラムとを搭載し、前記ファイルシステムにより前記情報を第1のフォーマットと第2のフォーマットにフォーマット変換して、第1の検索プログラムと第2の検索プログラムに提供する請求項26記載の検索サーバ計算機。

【請求項28】複数のファイルを保持する二次記憶装置を備え、暗号化機能を有するサーバ計算機と、復号化機能を有するクライアント計算機とが通信ネットワークで結合された電子商取引システムであって、サーバ計算機は、ファイルを編集する機能を有する編集プログラムと、暗号化プログラムと、該暗号化プログラムを変換プ

ログラムとして請求項1ー15いずれか1項記載のファイルフォーマット変換方法により変換元ファイルを変換先ファイルにフォーマット変換するファイルシステムとを備え、前記編集プログラムより入力された商取引情報を該ファイルシステムにおける変換元ファイルに格納し、該変換元ファイルを前記暗号化プログラムにより暗号化して変換先ファイルに格納し、該変換先ファイルを通信ネットワークを経由してクライアント計算機に送信するものであり、クライアント計算機は、前記編集プログラムと、復号化プログラムと、該復号化プログラムを変換プログラムとして請求項1ー15いずれか1項記載のファイルフォーマット変換方法により変換元ファイルを変換先ファイルにフォーマット変換するファイルシステムとを備え、該ファイルシステムにおける変換元ファイルに前記送信されたファイルの内容を格納し復号化プログラムにより該変換元ファイルを復号化し、復号化した変換元ファイルを前記ファイルシステムにおけるファイルフォーマット変換方法により変換先ファイルにフォーマット変換し、前記編集プログラムにより該変換先ファイルの内容を参照する、電子商取引システム。

【発明の詳細な説明】

【0001】

【発明の属する技術分野】本発明は、計算機システムに関し、特に、ユーザに対して複数のファイルフォーマットで情報を提供するファイルシステムのフォーマット変換方法に関し、さらに具体的には、World Wide Web（以下、WWWという）によって複数の計算機が複数のファイルフォーマットで情報交換を行う際に好適なファイルフォーマット変換方法とこれを用いたファイルシステム及び種々の情報処理システムに関する。

【0002】

【従来の技術】従来の技術を説明する前に、まず、いくつかの用語を説明する。

【0003】計算機システムにおいては、計算機の電源がオフになっても保持されるべきデータを格納するために「二次記憶装置」が用いられる。現在用いられている二次記憶装置の例としては、ハードディスク、フロッピーディスク、光磁気ディスクなどがある。

【0004】「ファイルシステム」は、二次記憶装置とユーザの応用プログラム（「アプリケーション」）の間のデータの交換を司るソフトウェアである。ファイルシステムが二次記憶装置のデータを操作する単位は「ファイル」と呼ばれる。一般に複数のファイルがひとつの二次記憶装置に保持され、それぞれのファイルは「ファイル名」で区別される。ファイル名には文字列がよく使用される。ファイルシステムは多くの場合オペレーティング・システム（OS）の一部としてユーザに提供されるが、ライブラリとOSの組み合わせ、あるいはライブラリのみで提供される場合もある。すなわち、例えばOSが提供するファイルシステムと、当該ファイルシステム

に拡張機能を付加するライブラリとを組み合わせたソフトウェアも、二次記憶装置とアプリケーションの間のデータの交換を司るソフトウェアであればファイルシステムと呼ぶ。

【0005】 応用プログラム（アプリケーション）が行なうことのできるファイルへの操作は、アプリケーションとファイルシステムとのアプリケーション・プログラミング・インターフェース（API）によって定義されている。特に、ファイルの入出力に関するAPIを「ファイル入出力API」と呼ぶ。ファイル入出力APIには、オープン（ファイル読み書きの準備）、読み出し（ファイルからアプリケーションへのデータの移動）、書き込み（アプリケーションからファイルへのデータの移動）、クローズ（ファイル読み書きの終了）が含まれる。また、ファイルとディレクトリの操作に関するAPIを「ファイル管理API」と呼ぶ。ファイル管理APIには、ファイル作成、ファイル名変更、ファイル削除が含まれる。ファイル入出力APIとファイル管理APIで可能となるファイルへの操作を総称して「ファイル操作」と呼ぶ。

【0006】 ファイル内のデータが持つ構造を「ファイルフォーマット」（または単に「フォーマット」）と呼ぶ。フォーマットの例としては、「改行文字で区切られた文字列の並び」、「タブ文字で区切られた項目群の並び」、「ワープロソフトAのファイル形式」、「動画のコマ（ある瞬間に表示される画像）の並び」などがある。ファイル名の最後の数文字を用いて、ファイルフォーマットを表現することがある。このファイル名の最後の数文字を「拡張子」と呼ぶ。

【0007】 以下、従来の技術について述べる。

【0008】 現在多くのアプリケーション（例えばワード・プロセッサ、表計算、スケジュール管理、電子メール、プログラミング・ツールなど）が用いるファイルは、それぞれ固有のフォーマットを持っている。特に、近年広く使われるようになったパーソナル・コンピュータでは、多くのファイルフォーマットが使われており、その数は200近くにもものぼる。また、インターネットの発展に伴って出現したWorld Wide Web（WWW）では、単なる文字だけでなく、静止画像、動画像、音声、コンピュータ・グラフィクスなど、多種多様なフォーマットが使用されている。

【0009】 一般にすべてのアプリケーションがすべてのファイルフォーマットをアクセス可能ではない。このため、ユーザがある情報のあるファイルフォーマットでファイルに格納したとしても、別のアプリケーションからその情報にアクセスするために、そのファイルを「フォーマット変換」を行う必要が生じることが多い。フォーマット変換の入力となるファイルを「変換元ファイル」、出力となるファイルを「変換先ファイル」と呼ぶ。

【0010】 フォーマット変換には、多種多様なフォーマットに関する多数の「変換プログラム」から、（1）ユーザの所望する変換を行う変換プログラム（または複数の変換プログラムの組合わせ）を選ぶ作業と、（2）選んだ変換プログラムに対して、変換元ファイルと変換先ファイルとを正しく与え、適切な時点で変換プログラムを実行する作業が必要となる。これらの作業は、アプリケーション上で行われるユーザの本来の作業とは関係ない。このため、フォーマット変換に関するユーザの負担をできる限り小さくすることが望まれる。特に上述のようにインターネットとWWWの利用が激増していることから、多種多様なフォーマットをできる限り簡単に扱える必要性が高まっている。

【0011】 これまでに主に上記（1）の作業を簡単化するために、以下の方法が提案されている。

【0012】 特開平6-187219号「プログラム間の自動データ共有方法」（以下公知例1と記す）では、利用するアプリケーションとファイルの名前をユーザが指示し、アプリケーション・データ対応表を検索して適切な変換プログラムを選択し、当該アプリケーションで当該ファイルを利用するために必要なフォーマット変換を行う。

【0013】 これにより、ユーザは上記（1）の作業、すなわち当該ファイルをアクセスするために使用すべき変換プログラムを選ぶ作業から解放される。一方（2）の作業、すなわちフォーマット変換をいつどのファイルに対して行うかを指定する作業は、依然としてユーザに課せられたままである。なぜなら、公知例1のユーザは、アプリケーション上で行われるユーザの本来の作業には必要ない、変換したいファイルをアプリケーション名とともに公知例1のシステムに与えるという作業を要求されるためである。

【0014】 また、WWWを含む現在の計算機システムでは、1つのアプリケーションが複数のファイルフォーマットを取り扱い、また1つのファイルフォーマットを取り扱うアプリケーションが複数存在する。このような状況ではアプリケーションとファイルの名前を指定しても変換方法が一意に定まらない。例えば、.tex、.ps、.pdfの3つの拡張子によって区別される3つのファイルフォーマットがあり、プログラムAが.psと.pdfを取り扱える場合を考える。ユーザがプログラムAとファイルfoo.texを指定しても、foo.texをfoo.psに変換すべきか、foo.pdfに変換すべきかは一意に決まらない。

【0015】 特開平9-69059号「ファイル形態変換システム」（以下公知例2と記す）では、変換元ファイル名と変換先ファイル名（または変換元ファイルを使用するアプリケーション名と変換先ファイルを使用するアプリケーション名）をユーザが指定すると、変換プログラムを選びまたは組み合わせて、変換元ファイルから

変換先ファイルを得るフォーマット変換を行う。これにより、ユーザは上記(1)の作業、すなわち当該ファイルをアクセスするために使用すべき変換プログラムを選びまたは組み合わせる作業から解放される。一方(2)の作業、すなわちフォーマット変換をいつどのファイルに対して行うかの指定は、ユーザが行う必要があり、この負担はユーザに課せられたままである。

【0016】また複雑なファイル処理過程を一括して自動化する方法としては、University of California, Computer Systems Research Group著の文献「4.4BSD User's Reference Manual」のMAKE(1)章(O'Reilly&Associates, Inc, 1994)に記載のソフトウェア(以下公知例3と記す)が知られている。公知例3はソース・プログラムからバイナリ・プログラムへの、一段階または多段階にわたるコンパイル作業を簡単化する方法である。公知例3も(1)の課題を解決するが、(2)の課題は解決しない。

【0017】

〔発明が解決しようとする課題〕上述のように、多くの提案にもかかわらず従来のフォーマット変換方法では、上記(2)の作業がユーザに与える負担に対する配慮がされていなかった。(2)の作業はさらに以下の2つの作業に分けることができる。

【0018】(2-a)どのファイルを変換元ファイルとし、どのファイルを変換先ファイルにするかを指定する作業。この指定を間違えると、変換先ファイルの内容がユーザ所望のものにならなかったり、他のファイルの内容を破壊する危険性がある。特にフォーマット変換以前には変換先ファイルが存在しない場合がある点に注意が必要である。

【0019】(2-b)フォーマット変換をいつ行うかを指定する作業。この指定を間違えると、アプリケーションが古い情報にアクセスする危険性がある。

【0020】加えて、従来のフォーマット変換方法では、以下の課題があった。

【0021】(i)変換元ファイルと変換先ファイルの間の一貫性をとる方法が用意されていなかったため、変換元ファイルと変換先ファイルにほぼ同時に書き込みがおこって、続くフォーマット変換によってどちらかの書き込みが失われる、などの危険性がある。

【0022】(ii)多数の変換先ファイルを保持するために、二次記憶領域が余分に必要となる。

【0023】(iii)変換元ファイルが何らかの理由(例えば変換元ファイルのあるアプリケーションが編集中心だったり、変換元ファイルを保持する二次記憶装置の電源がオフになっている等)で利用不能な間は、フォーマット変換を行うことができない。

【0024】従って本発明の目的は、複数のファイルフ

ォーマットが複数のアプリケーションから利用可能な現状を考慮した上で、上記の(1)とともに、(2-a)、(2-b)の作業にともなう課題に加えて上記(i)、(ii)、(iii)の課題を解決し、複数のファイルフォーマットを利用する際のユーザの手間をできる限り減らすことである。

【0025】

〔課題を解決するための手段〕(2-a)の作業に伴う課題は、変換元ファイルと変換先ファイルがそれぞれ多数あり、かつこれらの間の関係があらかじめ与えられていないことに起因している。このために、どのファイルをどのファイルへフォーマット変換するかをユーザがその都度指示する必要があったが、本発明では、変換元ファイルと変換先ファイルと変換プログラムとの間の対応関係(1段階でも、多段階でも良い)をファイルシステム中に保持する。この手段によって、フォーマット変換方法が変換先ファイルのみから変換元ファイルと変換プログラムとを得ることを可能とした。同時に、変換元ファイルのみから1つ以上の変換先ファイルと、対応する1つ以上の変換プログラムを得ることも可能とした。

【0026】また、変換先ファイルが変換以前には存在しない場合に対応するために、変換元ファイルのファイル名から変換先ファイルのファイル名を得るファイル名変換方法をファイルシステム中に保持する。通常なら変換先ファイルはフォーマット変換の結果としてできるが、本発明では変換先ファイルを指定することによりフォーマット変換を開始することができるようにしてユーザの手間を減らす。すなわちフォーマット変換と変換先ファイルがいわゆる「鶏と卵」の状態になる。この問題を、ファイル名変換方法を備えることにより解決し、フォーマット変換を行う以前に変換先ファイルをユーザに提供することを可能にした。

【0027】(2-b)は、アプリケーションとフォーマット変換方法の連携をとる手段がないこと、すなわちアプリケーションがどのファイルをいつ、どのように操作するかをフォーマット変換方法が知る術がなかったことに起因する。このため従来は、フォーマット変換の起動作業を、アプリケーションによる作業とは別にユーザが行わざるを得なかった。これに対し本発明では、ユーザによるファイル入出力APIの発行を契機として、ファイルシステムから変換プログラムを起動して実行する。

【0028】以上の2点を備えることにより、本発明のフォーマット変換方法は、アプリケーションがどのファイルを操作の対象とし、アクセスが書き込みか読み出しかを知ることができる。すなわち、ユーザはアプリケーション上で行うユーザの本来の作業を行えばよく、その過程で必要となる様々なフォーマット変換(1段階でも、多段階でも良い)はユーザの関与なしで行われる。特にファイル操作を契機として変換プログラムを実

行するため、フォーマット変換のタイミングを指定する必要がなく、ユーザは常に最新の変換先ファイルが利用可能となる。

【0029】加えて本発明では、上記(i)、(ii)、(iii)の開題を解決するため、それぞれ以下(I)、(II)、(III)の手段を設ける。

【0030】(I)変換元ファイルに対するファイル入出力APIの実行と、変換先ファイルに対するファイル入出力APIの実行との排他制御を行う。すなわち、いずれか一方が実行されている間は他方を実行しないよう制御する。これにより、変換元ファイルと変換先ファイルの一貫性を保持できる。

【0031】(II)多数の変換先ファイルを保持することによる二次記憶領域の無駄を解決するために、変換先ファイルを必要に応じて消去する。

【0032】(III)変換元ファイルが利用不能な間もフォーマット変換を行うことを可能にするために、中間ファイルを導入し、変換元ファイルから中間ファイルへ、中間ファイルから変換先ファイルへの2段階の変換を行う。

【0033】ファイルシステムはファイルに対するあらゆる操作要求を受け取るので、ファイル入出力APIを契機とした変換や、排他制御、二次記憶領域の無駄の管理、中間ファイルの管理が実現できる。さらにファイルシステムは、多くのアプリケーションによって共有されているので、上記解決手段をファイルシステムが備えることによって多くのアプリケーションがこの発明の利益を享受することができる。

【0034】

【発明の実施の形態】以下、本発明の実施の一形態を、図面を参照しながら説明する。なお簡単のため、本明細書中では以下に述べる発明の実施の形態を「第1の実施例」と呼ぶことにする。

【0035】全体構成

図1と図2を用いて第1の実施例の構成を説明する。なお、図1において、矢印付きの太い線(154—160、162—166)は主要なデータの流れを、また矢印付きの細い線(150—153、161、167)は主要な制御の流れを表わす。

【0036】計算機10は、いわゆるパーソナル・コンピュータ、ワークステーション、並列計算機、大型計算機等、任意のコンピュータでよい。二次記憶装置11はファイルを保存する記憶装置である。二次記憶装置11には不揮発性記憶装置(磁気ハードディスク、光ディスク等)がよく用いられるが、特別な場合には揮発性記憶装置(メインメモリ、キャッシュメモリ等)が用いられる場合もある。計算機10と二次記憶装置11の間の結合には様々な形態が考えられるが、計算機10にファイルを提供できればよく、特定の結合形態を必要としない。例えば、計算機10が占有する通信線による結合、

複数の計算機によって共有されるネットワークを介した結合、他の計算機経由の結合等が代表的な例として考えられる。また、計算機10に対して二次記憶装置11を1つ以上用いても差し支えない。

【0037】ファイルシステム100は計算機10上で動作するソフトウェアである。ファイルシステム100はデータをファイルと呼ぶ単位で管理、保存する。1つ以上のファイルが二次記憶装置11に保持され、それぞれのファイルはファイル名で区別される。

【0038】アプリケーション101、102はユーザの実行する応用プログラムである。多くのアプリケーションはファイルを操作するためにファイルシステム100とのやり取りを行う。アプリケーション101、102がファイルシステム100に対して要求するファイル操作は、ファイル入出力APIとファイル管理APIを含む。

【0039】変換元ファイル130は、フォーマット変換の入力となるファイルである。変換先ファイル群131、131'、…は、フォーマット変換の出力となるファイルである。なお、第1の実施例では説明を明瞭にするために変換元ファイル130と変換先ファイル群131、131'、…を別々のファイルにしてあるが、1つのファイルが変換元ファイル130かつ変換先ファイル131であっても差し支えない。また図1に示した変換元ファイルと変換先ファイルは、それぞれ複数存在して差し支えない。

【0040】変換プログラム群103、103'、…はファイルシステム100が実行するフォーマット変換のためのプログラム群である。本実施例では、変換プログラム群103、103'、…はファイルシステム100の外に置かれてフォーマット変換を実行しているが、ファイルシステム100に内蔵して、ファイルシステム100内でフォーマット変換を実行しても差し支えない。

【0041】変換表120は、変換元ファイルと変換先ファイルの組を、変換プログラムに対応づける表である。図2に示すように、変換表120は1つ以上の変換表エントリ200からなり、1つの変換表エントリ200が1通りのフォーマット変換に対応する。変換表エントリ200はさらに、変換元フォーマット201、変換先フォーマット202、変換プログラム203を含む。変換元フォーマット201は変換元ファイルのフォーマット、変換先フォーマット202は変換先ファイルのフォーマットである。変換プログラム203は変換元フォーマット201のフォーマットを持つ変換元ファイルから、変換先フォーマット202のフォーマットを持つ変換先ファイルへの変換を行うプログラムで、その名前(および必要ならば起動時引数)を持つ。

【0042】名前空間表121は、ファイルシステム100が管理するファイル群のファイル名と、ファイルシステム内部で利用されるファイルIDを対応づける。図

2に示すように、名前空間表121は1つ以上の名前空間表エントリ210からなり、1つの名前空間表エントリ210が1つのファイルに対応する。名前空間表エントリ210はさらに、ファイル名211とファイルID212を含む。ファイル名211は名前空間表エントリ210のファイルのファイル名、ファイルID212は当該ファイルのファイルIDである。ファイルIDは、ファイルシステム100内部で使用するファイルの番号で、ファイルIDとファイルは一対一に対応する。

【0043】ファイル表122は、ファイルシステム100が管理するファイルの諸属性を保持する。特に、ファイルのフォーマットと、(もし当該ファイルが変換先ファイルであれば)変換元ファイルをこの表に保持する。図2に示すように、ファイル表122は1つ以上のファイル表エントリ220からなり、1つのファイル表エントリ220が1つのファイルに対応する。ファイル表エントリ220はさらに、ファイルID221、フォーマット222、変換元ファイルID224、トークンID225、ファイル内容226を含む。ファイルID221は当該ファイルのファイルID、フォーマット222は当該ファイルのファイルフォーマット、タイムスタンプ223は当該ファイルに最後に書き込みがあった時刻、変換元ファイルID224は(もし当該ファイルが変換先ファイルであれば)変換元ファイルのファイルID、トークンID225は当該ファイルに対して割り当てられトークン表123で管理されるトークンIDである。ファイル内容226は当該ファイル本体(すなわちファイルのデータを格納する部分である。図1の変換元ファイル130や変換先ファイル群131、131'、...)および当該ファイルの属性である。

【0044】なお、変換元フォーマット201、変換先フォーマット202、フォーマット222に格納すべきフォーマットの表現方法にはいくつか考えられるが、本発明はこの方法には依存しない。例えば、拡張子がファイルのフォーマットを表わしている計算機システムでは、拡張子を格納することができる。また、ファイル内容226の一部または全部によってフォーマットが決定される場合もあるが、この場合にはフォーマット毎に異なるフォーマット名を決めておき、フォーマット名を格納することができる。本実施例では、変換元フォーマット201、変換先フォーマット202、フォーマット222に拡張子を格納するものとする。

【0045】トークン表123は、トークンを管理する表である。1つのトークンは、ある変換元ファイルと、当該変換元ファイルから変換可能な変換先ファイル群からなる一群のファイルに対して割り当てられる。図2に示すように、トークン表123は1つ以上のトークン表エントリ230からなり、1つのトークン表エントリ230が1つのトークンに対応する。トークン表エントリ230はさらにトークンID231とファイルID23

2からなる。トークンID231はトークンを一意に識別する番号、ファイルID232は、現在トークンを保持しているファイルのファイルIDである。モード233は、当該ファイルの現在のオープン・モードを表わす。

【0046】削除候補表124は、変換先ファイルを列挙した表である。二次記憶装置の空き領域を確保するなどの目的で、変換先ファイルを削除する際に用いる。図2に示すように、削除候補表124は1つ以上の削除候補表エントリ240からなり、1つの削除候補表エントリ240が1つの削除可能なファイルに対応する。削除候補表エントリ240は削除可能なファイルのIDであるファイルID241を持つ。

【0047】図1のフォーマット変換制御部110は、アプリケーション101、102から発行されるファイル入出力APIおよびファイル管理APIの呼び出し(APIコール)を受け取り、フォーマット変換を制御する部分である(制御の詳細は後述する)。

【0048】設定プログラム104は、ファイルシステム100によるフォーマット変換の動作をユーザが設定、変更するためのフォーマット変換設定APIを、フォーマット変換設定部111に対して発行するプログラムである。フォーマット変換設定部111は、ユーザからのフォーマット変換設定APIを受けて(161)、変換表120とファイル表122を変更する。フォーマット変換設定APIは、変換表120の各エントリ200、およびファイル表122の各エントリ210を参照し、変更する(162)。

【0049】なお、変換表120、ファイル表122、名前空間表121、トークン表123および削除候補表124はいずれも、主記憶上か二次記憶上のいずれか片方または両方にあつて差し支えない。また、変換表120、ファイル表122、名前空間表121、トークン表123および削除候補表124はいずれも、ファイルシステム100の外に保存されても差し支えない。例えば、ファイルシステム100と別のプログラムがあつて、変換表120、ファイル表122、名前空間表121、トークン表123および削除候補表124の一部または全部に対する参照と更新を提供していて、ファイルシステム100が当該プログラム経由で各表にアクセスしても差し支えない。

【0050】動作の説明

以下、第1の実施例の動作を説明する。本発明が特徴を発揮する以下の3つの場面を想定し、順に説明する。

(1) 第1のアプリケーション101が変換元ファイル130を作成する。(2) 第1のアプリケーション101が変換元ファイル130に対して読み書きを行う。

(3) 二次記憶装置11の空き領域が不足した際に、変換先ファイル131を消去する。なお、ここでは説明を簡単にするため、第1のアプリケーション101と第2

のアプリケーション102をとしているが、この数および種類は一例にすぎない。アプリケーションの数および種類は1つでも、3つ以上でも差し支えない。

【0051】(1)アプリケーションによるファイルの作成

第1のアプリケーション101または第2のアプリケーション102は、ファイルシステム100のファイル作成APIを生成対象の第1のファイルの第1のファイル名とともに起動することにより、ファイル作成をファイルシステム100に依頼する(150、152)。この際、ファイルシステム100は図3に示す処理を行う。

【0052】まず、第1のファイル名を名前空間表121に登録する(155、ステップ301)。具体的には、新たな名前空間表エントリ210を名前空間表121に割り当て、ファイルID212に他のファイルにまだ割り当てられていない第1のファイルIDを格納し、ファイル名211に第1のファイル名を格納する。

【0053】次に、第1のファイルをファイル表122に登録する(156、ステップ302)。具体的には、新たなファイル表エントリ220をファイル表122に割り当て、ファイルID221に第1のファイルIDを、ファイル名の拡張子から決定したフォーマット222に第1のファイルの第1のフォーマットを、タイムスタンプ223に現在時刻を、変換元ファイルID224に「空」を、トークンID225にまだどのトークンにも割り当てられていない第1のトークンIDを、ファイル内容226のファイル本体(すなわち変換元ファイル130)に「空」をそれぞれ格納する(159)。なお、ファイル内容226が決定しなければファイルのフォーマットがわからない計算機システムの場合には、例えばフォーマット222に「空」をいれておき、後で第1のファイルに対する書き込み等によってファイル内容226が得られてから、フォーマット222を格納する。

【0054】次に、第1のファイルに対応するトークンをトークン表123に登録する(157、ステップ303)。具体的には、新たなトークン表エントリ230をトークン表123に割り当て、トークンID231に第1のトークンIDを、ファイルID232に「空」を格納する。

【0055】次に、1つ以上の第1の変換先ファイル名を第1のファイル名と変換表120から生成する(154、ステップ304)。具体的には、変換表120に格納されている変換表エントリ200のうち、変換元フォーマット201が第1のフォーマットに等しい1つ以上の第1の変換表エントリを検索する。そして、第1の変換表エントリのそれぞれについて、第1のファイル名から拡張子を取り去って、代りの拡張子として変換先フォーマット202をつけたものを変換先ファイル名とする。これが本実施例におけるファイル名変換方法であ

る。

【0056】次に、生成した上記変換先ファイル名のそれぞれについて(ステップ305の判定がNの間)、ステップ306とステップ307の処理を行う。具体的には、ステップ301と同様の方法で上記変換先ファイル名を名前空間表121に登録し(155、ステップ306)、変換先ファイルをファイル表122に登録する(156、ステップ307)。変換先ファイルのファイル表122への登録では、新たなファイル表エントリ220をファイル表122に割り当て、ファイルID221に第1のファイルIDを、ファイル名の拡張子から決定したフォーマット222に第1のファイルの第1のフォーマットを、タイムスタンプ223に「空」を、変換元ファイルID224に第1のファイルIDを、トークンID225に第1のトークンIDを、ファイル内容226のファイル本体(すなわち変換先ファイル群131、131'、…のいずれか)に「空」を格納する(160)。また、変換先ファイルがさらに変換元ファイルとなりうる場合、すなわち多段階の変換が可能な場合、変換先ファイルのファイル名をさらにファイル名変換方法で変換し、得られたファイル名およびファイルを本段落の手順で名前空間表121とファイル表122にそれぞれ登録する。

【0057】ステップ304で生成したすべての変換先ファイル名を処理したら(ステップ305の判定がY)、残りのファイル生成操作を行う(ステップ308)。この操作の中で、例えば二次記憶装置11へのディスクブロックの割り当て、ファイルの所有者、アクセス権、生成時刻等の属性の設定などを行う場合があるが、これらは既に公知であり、本発明の特徴とも直接関係ないのでこれ以上説明しない。

【0058】以上のファイル作成APIが終了した時点で、ファイルシステム100は第1のアプリケーション101または第2のアプリケーション102に対して、結果を返す(151、153)。この結果はファイルID(またはファイルIDを間接に指す番号)である。

【0059】以上が第1の実施例において新たなファイルを生成する際の動作である。上述のように、第1の実施例のファイルシステムは、変換元ファイルのファイル名から変換先ファイルのファイル名を得るファイル名変換方法をファイルシステムが持つ。これにより、ユーザがファイル作成APIによって直接作成を依頼した第1のファイルが作成されるに加えて、第1のファイルを変換元ファイルとする1つ以上の変換先ファイルがファイルシステムにより生成される。

【0060】なおこの実施例では、変換元ファイルの作成操作を契機として変換先ファイルのファイル名を生成したが、本発明はこの契機に限定されるものではない。別の契機として、例えばユーザがディレクトリの表示操作または検索操作を用いることが可能である。変換先フ

ファイルのファイル名が生成されるディレクトリに「ファイル名生成を行う必要あり」の印をつけておき、後に当該ディレクトリを表示または検索する際に、実際のファイル名生成を行う。こうすることで、ファイル名生成の処理を遅延し、変換元ファイルの作成操作をより高速にすることが可能である。また、「ファイル名生成を行う必要あり」の印がついたファイルを、一定時間後、または一定時間毎に探し出して、変換先ファイルのファイル名を生成しても良い。

【0061】(2) アプリケーションによるファイルへの読み書き

アプリケーション群(101、102)による第1のファイルへの書き込みは、オープンAPIを用いて書き込みモードで第1のファイルをオープンし、書き込みAPIまたは読み出しAPIを用いてデータを読み書きし、クローズAPIを用いて第1のファイルをクローズする、という手順で行う。なお、計算機システムによってはオープンAPIやクローズAPIがない場合も考えられる。このような場合には、読み出しAPI、書き込みAPIが毎回、オープンとクローズをそれぞれのAPI処理の最初と最後に行われるとみなして本発明を適用することができる。

【0062】オープンAPIの動作を、図4を用いて説明する。第1の実施例のオープンAPIは、第1のファイルの第1のファイル名とオープンモード(読み出し、書き込み、または両方)を伴ってアプリケーション群(101、102)から呼び出される(150、152)。

【0063】まず、第1のファイルと変換可能なファイル群の読み書きの一貫性を保持するために、トークンを取得する(ステップ401)。具体的には、名前空間表121の中から、ファイル名211が第1のファイル名に等しい名前空間表エントリ210を検索し(154)、当該第1の名前空間表エントリのファイルID212に等しいファイルID221を持つファイル表エントリ220を検索し(156)、当該第1のファイル表エントリのトークンID225に等しいトークンID231を持つトークン表エントリ230をトークン表123の中から検索する(157)。そして、当該第1のトークン表エントリのファイルID232が「空」になるまで待ち、ファイルID232に第1のファイル表エントリのトークンID225を格納する。モード233には、オープンモードを格納する。本実施例では、トークンで変換可能なファイル群へのファイル操作をすべて直列化しているが、2つ以上の書き込みAPI(または2つ以上の書き込みオープンAPI)を禁止するトークン取得・解放アルゴリズムを使用しても差し支えない。

【0064】次に、第1のファイルが変換先ファイルか(Y)否か(N)を判定する(ステップ402)。具体的には、前記第1のファイル表エントリの変換元ファ

イルID224が「空」でなければ、第1のファイルは変換先ファイルである。

【0065】第1のファイルが変換先ファイルでなければ(ステップ402の判定がN)、後述のステップ408に制御を移す。一方、第1のファイルが変換先ファイルであれば(ステップ402の判定がY)、第1のファイルの第1の変換元ファイルを決定する(ステップ403)。具体的には、第1のファイル表エントリの変換元ファイルID224が第1の変換元ファイルのファイルIDである。当該ファイルIDに等しいファイルID221を持つファイル表エントリ220を検索し、当該第2のファイル表エントリとする。

【0066】次に、変換先ファイルである第1のファイルが第1の変換元ファイルの最新の内容を反映しているかを確認する。第1のファイル表エントリのタイムスタンプ223が「空」、または、「第1のファイル表エントリのタイムスタンプ223<第1のファイル表エントリのタイムスタンプ223」であれば(ステップ404の判定がY)、最新の内容を反映していない。

【0067】この場合にはまず、変換元ファイルをオープンし(ステップ405)、変換プログラムを選択して実行する(ステップ406)。具体的には、変換表120の中から、変換先フォーマット202が第1のファイル表エントリのフォーマット222に等しく、かつ変換元フォーマット201が第2のファイル表エントリのフォーマット222に等しい変換表エントリ200を検索し(154、156)、第2のファイル表エントリのファイルID221で示すファイルをオープンし(ステップ405)する。そして、当該第1の変換表エントリの変換プログラム203を起動する(167、(ステップ406))。この際、変換プログラム203への入力として第2のファイル表エントリのファイル内容226のファイル本体を与え(163)、変換プログラム203の出力を第1のファイル表エントリのファイル内容226のファイル本体に格納する(164)。この際、第2のファイル表エントリのファイル内容226のファイル本体がファイルシステム100中になければ、二次記憶装置11から取出す(165)。また変換プログラム203終了後、第1のファイル表エントリのファイル内容226のファイル本体を、二次記憶装置11に書き出す(166)。なお、第1のファイル表エントリのファイル内容226のファイル本体を二次記憶装置11に書き出すことは、必ずしも必要ではない。変換が終わったら、第2のファイル表エントリのファイルID221で示すファイルをあとで述べる方法でクローズする(ステップ407)。

【0068】そして、変換先ファイルのタイムスタンプを現在時刻に設定する(156、ステップ408)。具体的には、第1のファイル表エントリのタイムスタンプ223に現在時刻を格納する。

【0069】さらに、ステップ402で変換先ファイルのタイムスタンプが「空」であったならば、変換先ファイルを削除候補表124に登録する(158、ステップ409)。具体的には、削除候補表124に新たな削除候補表エントリ240を割り当て、当該削除候補表エントリのファイルID241に第1のファイル表エントリのファイルID221を格納する。

【0070】一方ステップ404の判定がNなら、第1のファイルは第1の変換元ファイルの最新の内容を反映しているので、ステップ410で残りのファイル・オープン操作を行う。この操作の中で、ファイル・ポインタの初期化、主記憶上の読み書き用バッファの割り当て、ファイルアクセス権限の確認などを行う場合があるが、これらは既に公知であり、本発明の特徴とも直接関係ないのでこれ以上説明しない。

【0071】以上のオープンAPIが終了した時点で、ファイルシステム100は当該オープンAPIを呼び出したアプリケーションに対し、当該オープンAPIが完了したことを通知する(151、153)。

【0072】次に、書き込みAPIの動作を説明する。書き込みAPIは、従来のファイルシステムと同様、アプリケーションがファイル名またはファイルID(またはファイルIDを間接に指す番号)を指定して、ファイルシステム100にデータの書き込みを依頼する(150、152)。この際ファイルシステム100は、与えられたものがファイル名であれば上述のオープンAPIと同様名前空間表121からファイルIDを得(155)、また与えられたのがファイルIDであればそのファイルIDを用いる。そして、当該ファイルIDから当該ファイルIDに対応するファイル表エントリ220を得て、当該ファイル表エントリのファイル内容226のファイル本体に、与えられたデータを書き込む(156)。以上の書き込みAPIが終了した時点で、ファイルシステム100は当該書き込みAPIを呼び出したアプリケーションに対し、当該書き込みAPIが完了したことを通知する(151、153)。

【0073】次に、読み出しAPIの動作を説明する。読み出しAPIは、従来のファイルシステムと同様、アプリケーションがファイル名またはファイルID(またはファイルIDを間接に指す番号)を指定して、ファイルシステム100にデータの読み出しを依頼する(150、152)。この際ファイルシステム100は、与えられたものがファイル名であれば上述のオープンAPIと同様名前空間表121からファイルIDを得(155)、また与えられたのがファイルIDであればそのファイルIDを用いる。そして、当該ファイルIDから当該ファイルIDに対応するファイル表エントリ220を得て、当該ファイル表エントリのファイル内容226のファイル本体から、与えられたデータを読み出す(156)。以上の読み出しAPIが終了した時点で、ファ

イルシステム100は当該読み出しAPIを呼び出したアプリケーションに対し、当該読み出しAPIが完了したことを通知する(151、153)。

【0074】次に、クローズAPIの動作を図5を用いて説明する。

【0075】第1の実施例のクローズAPIは、オープンAPIでオープンされた前記第1のファイルの第1のファイルIDを伴ってアプリケーション群(101、102)から呼び出される(150、152)。

【0076】第1のファイルのオープンが書き込みモードであって、第1のファイルが変換元ファイルであるかをチェックする(ステップ501)。オープンが書き込みモードかどうかは、以下の方法でチェックする。第1のファイルIDに等しいファイルID221を持つファイル表エントリ220を検索し(156)、当該第1のファイル表エントリのトークンID225に等しいトークンID231を持つトークン表エントリ230をトークン表123の中から検索する(157)。そして、当該第1のトークン表エントリのモード233が書き込みモードかチェックする。また、第1のファイル表エントリの変換元ファイルID224が「空」ならば、第1のファイルが変換元ファイルであり、「空」以外ならば第1のファイルは変換先ファイルである(156)。

【0077】ステップ501がYなら、ステップ502以下を実行する。具体的には、第1のファイル表エントリのタイムスタンプ223に現在時刻を格納する(156)。一方ステップ501がNなら、ステップ503に制御を移す。

【0078】次に、トークンを解放する(ステップ503)。具体的には、第1のトークン表エントリのファイルID232とモード233に「空」を格納する(157)。

【0079】最後に、ステップ504で残りのファイル・クローズ操作を行う。この操作の中で、主記憶上の読み書き用バッファの解放、ファイル内容の二次記憶装置11への書き出しなどを行う場合があるが、これらは既に公知であり、本発明の特徴とも直接関係ないのでこれ以上説明しない。

【0080】以上のクローズAPIが終了した時点で、ファイルシステム100は当該クローズAPIを呼び出したアプリケーションに対し、当該クローズAPIが完了したことを通知する(151、153)。

【0081】以上のように第1の実施例のファイルシステムでは、主に変換表120に変換元ファイルと変換先ファイルと変換プログラムとの間の対応関係(1段階でも、多段階でも良い)をファイルシステム中に保持して、どのファイルをどのファイルへ変換するかをユーザがその都度指示する必要をなくしている。

【0082】また、第1の実施例はファイルへの読み書きに際して必ずアプリケーションが行う動作(オープン

APIとクローズAPI、読み出しAPIや書き込みAPIでも差し支えない)を、フォーマット変換の契機としても用いる。これにより、第1の実施例のフォーマット変換方法は、アプリケーションの動作をファイルシステムを通じて逐一知り、対応する動作を行うことができる。翻ってアプリケーションは、フォーマット変換方法の動作をまったく意識することがないが、ユーザの作業を進めるために必要な変換は順次行われる。

【0083】加えて第1の実施例では、主にトークン表123とタイムスタンプ223を用いて、変換元ファイルの最新の内容を変換先ファイルに反映させ、かつ、アプリケーションが変換元ファイルと変換先ファイルに対して同時に読み書きを要求した場合でも、これらを調停して変換元ファイルと変換先ファイルの一貫性を保持する。

【0084】(3)変換先ファイルの消去
二次記憶装置11の空き領域が切迫した場合、変換先ファイル群131、131'、…を消去することができる。これは以下の手順で行う。

【0085】ファイルシステム100は定期的に二次記憶装置11の空き領域の監視を行い、空き領域の量が一定以下になったら、削除候補表124に含まれる削除候補表エントリ240のそれぞれについて(158)、以下の処理を行う。

【0086】削除候補エントリのファイルID241と等しいファイルID221を持つファイル表エントリ220をファイル表122から検索する(156)。そして、当該ファイル表エントリのトークンID225と等しいトークンID231を持つトークン表エントリ230をトークン表123から検索し(157)、当該トークン表エントリのファイルID232が当該ファイル表エントリのファイルID221と異なっていれば、当該ファイル表エントリのファイル内容226のファイル本体を「空」にし、タイムスタンプ223に「空」を格納する(156)。

【0087】以上の操作によって、二次記憶装置11の空き領域が切迫した場合に、現在使用されていない変換先ファイルのファイル内容をすべて削除し、空き領域を増やすことができる。変換先ファイルのファイル内容を削除したとしても、新たにユーザが変換先ファイルを参照した場合には、上述のようにフォーマット変換がユーザの関与なしに起動され、当該変換先ファイルのファイル内容は再び充填される。このため、ユーザは変換先ファイルのファイル内容が削除されていたことをまったく意識せずに作業を進めることができる。

【0088】フォーマット変換のタイミング

本実施例では、オープンAPIとクローズAPIの処理中にフォーマット変換を起動しているが、APIを契機として、他の起動タイミングでフォーマット変換を行っても差し支えない。

【0089】例えば、変換元ファイルの書き込みAPIのあと、一定時間経過後にフォーマット変換を行ってもよい。例えばアプリケーションが書き込みAPIを行った時点で、ファイルシステム100がタイマーをT1秒後(T1は所定の秒数)に設定して、アプリケーションに制御を移す。そして、T1秒以内に続く書き込みAPIが発行されれば、再びタイマーをT1秒後に設定し直す。一方、T1秒以内に続く書き込みAPIが発行されなければ、フォーマット変換を行う。このようにすると、書き込みAPIの発行が連続して行われる間はフォーマット変換を控え、書き込みAPIの発行が止んでしばらくしたところで、フォーマット変換を行って、変換先ファイル群に最新の内容を伝播する、という動作ができる。

【0090】また、第1の実施例ではオープンAPIの発行に同期してフォーマット変換を行っていたが、上記の書き込みAPIと同様に、クローズAPIの処理後しばらくしてから、ファイルシステム100が自発的にフォーマット変換を行うことも可能である。こうすれば、オープンAPIの実行時にはフォーマット変換を行わなくてもよい場合が増えるので、オープンAPIがより高速に行えるようになる。自発的なフォーマット変換のタイミングとしては、クローズAPIのT2秒後(T2は所定の秒数)、通常業務が終了する深夜S時から(Sは適当な時刻)、計算機10のCPU負荷がL以下になった時点(Lは適当なCPU負荷指標)で、などの選択肢が考えられ、どれも有効である。これらの自発的なフォーマット変換は、API発行を契機としたフォーマット変換と併用することで、変換先ファイルが最新であることを保証するのが望ましい。

【0091】第2の実施例：2段階変換

第1の実施例と異なる本発明の形態を、図面を参照しながら説明する。以下に述べる発明の実施の形態を「第2の実施例」と呼ぶことにする。

【0092】全体構成

図6を用いて第2の実施例の構成を説明する。第2の実施例は、第1の実施例の主要部分を、分散システム(2つ以上の計算機がネットワークで結合された計算機システム)に拡張したものである。

【0093】なお、図6に示した計算機、ネットワーク、アプリケーション、変換プログラム、変換元ファイル、変換先ファイルの数と構成は、一例に過ぎず本発明はこれに限定されるものではない。また、図6はネットワーク12を介して2つの計算機10、10'が結合した構成をとっているが、第2の実施例で行っている処理を、ひとつの計算機上で実現しても差し支えない。

【0094】計算機10'は第1の実施例で説明した計算機10と同様、いわゆるパーソナル・コンピュータ、ワークステーション、並列計算機、大型計算機等、任意のコンピュータでよい。図6には二次記憶装置を図示し

ていないが、計算機10も計算機10'も二次記憶装置を持っていてよい。

【0095】ネットワーク12は、計算機10と計算機10'、または他の計算機を結合する。ここで、ネットワーク12はある団体（企業や学校や類似の団体）の全体や一部門でよく使用されるLANやWAN（イントラネットと呼ばれる場合がある）でもよく、また地理的に分散した複数の地点を結合するWANの一部または全部でもよい。すなわち、ネットワーク12はインターネットの一部または全部でよい。インターネットは、アメリカ合衆国を中心に発展したコンピュータ・ネットワークである。通信プロトコルとして主にTCP/IP（Transmission Control Protocol/Internet Protocolの略）を用いる。またネットワーク12は、計算機間結合網や並列計算機内部のプロセッサ要素間の結合網でもよい。

【0096】ファイルシステム100'は第1の実施例で説明したファイルシステム100と同じ機能を持つソフトウェアである。フォーマット変換制御部110'はフォーマット変換制御部110と、それぞれ同じ機能を持つソフトウェアであるが、以下に述べるようにクローズAPIの動作が第1の実施例とは異なる。

【0097】ファイルシステム100とファイルシステム100'はそれぞれ変換表120、ファイル表122、名前空間表121、トークン表123、削除候補表124を持つ（図6には示していない）。

【0098】動作の説明

以下、第2の実施例の動作を説明する。第2の実施例の特徴として、アプリケーション101が変換元ファイル130を作成して書き込み、アプリケーション102が変換元ファイル130から変換された変換先ファイル131を読み出す場合を想定して説明を行う。

【0099】アプリケーション101は、ファイルシステム100のファイル作成APIを生成対象の第1のファイルの第1のファイル名とともに起動することにより、変換元ファイル130の作成をファイルシステム100に依頼する（150）。この際、ファイルシステム100は第1の実施例ですでに説明した処理を図3のフローチャートに従って行う。ここで変換元ファイル130は、図6に示すように変換プログラム603および中間ファイル130'に対応づけられている。フォーマット変換制御部110はオープンAPIの途中で中間ファイルとして用いる中間ファイル130'を生成するためにフォーマット変換制御部110'と通信を行い（650）、フォーマット変換制御部110'が中間ファイル130'を生成する。

【0100】次に、アプリケーション101がファイルシステム100に対して、変換元ファイル130をオープンし、書き込みを行い、クローズする（150）。この一連の操作で変換元ファイル130のファイル内容が

格納される。変換元ファイル130のオープンAPIは図4を用いてすでに説明した手順で行う。また、変換元ファイル130の書き込みAPIも、第1の実施例ですでに説明した手順で行う。続くアプリケーション101へのクローズAPIの処理は、フォーマット変換制御部110が図7にしたがって行う。以下、この手順を説明する。

【0101】第2の実施例の変換元ファイル130のクローズAPIは、オープンAPIでオープンされた変換元ファイル130の第1のファイルIDを伴ってアプリケーション群（101、102）から呼び出される（150）。クローズの対象となっている第1のファイルのオープンが書き込みモードであって、第1のファイルが変換元ファイルであるか（Y）否か（N）を検査する

（ステップ701）。オープンが書き込みモードかどうかは以下の方法で検査する。第1のファイルIDに等しいファイルID221を持つファイル表エントリ220を検索し、当該第1のファイル表エントリのトークンID225に等しいトークンID231を持つトークン表エントリ230をトークン表123の中から検索する。そして、当該第1のトークン表エントリのモード233が書き込みモードかを検査する。また、第1のファイル表エントリの変換元ファイルID224が「空」ならば、第1のファイルが変換元ファイルであり、「空」以外ならば第1のファイルは変換先ファイルである。ステップ701の判定がYなら、ステップ702から707を実行する。一方、ステップ701の判定がNなら、ステップ703とステップ704を実行する。

【0102】ステップ702では、第1のファイル表エントリのタイムスタンプ223に現在時刻を格納する（156）。

【0103】次に、トークンを解放する（ステップ703）。具体的には、第1のトークン表エントリのファイルID232とモード233に「空」を格納する（157）。

【0104】次に、ステップ704で残りのファイル・クローズ操作を行う。この操作の中で、主記憶上の読み書き用バッファの解放、ファイル内容の二次記憶装置11への書き出しなどを行う場合があるが、これらは既に公知であり、本発明の特徴とも直接関係ないのでこれ以上説明しない。

【0105】次に、ステップ705で、第1のファイルに対する変換先ファイルを決定する。具体的には、ファイル表122の中で変換元ファイルID224が第1のファイルIDに等しいファイル表エントリ220を検索する。当該ファイル表エントリが変換先ファイルのファイル表エントリである。

【0106】次に、ステップ706で、当該変換先ファイルを読み出しモードでオープンする。このステップの処理は図4ですでに説明したものである。

【0107】次に、ステップ707で、第1のファイルをクローズする。このステップの処理は図5ですでに説明したものである。

【0108】以上のクローズAPIが終了した時点で、ファイルシステム100は当該クローズAPIを呼び出したアプリケーションに対し、当該クローズAPIが完了したことを通知する(151)。特にステップ706で、変換先ファイルをオープンしている点が、このオープンAPI処理の特徴である。このオープンによって、変換プログラム603によって変換元ファイル130の最新情報が中間ファイル130'に伝播する。加えてこのオープンによって、中間ファイル130'を変換元ファイルとした変換先ファイルのファイル名が名前空間表121に登録される。

【0109】次に、アプリケーション102がファイルシステム100'に対して、変換先ファイル131をオープンし、読み出しを依頼する(152)。この時、ファイルシステム100'は図4にしたがって、第1の実施例ですでに説明した処理を行う。ただ第2の実施例では、変換先ファイル131は変換プログラム103と中間ファイル130'に対応づけられている。このため、変換先ファイル131のオープンAPI操作は、中間ファイル130'を入力変換先ファイル131を出力として変換プログラム103を起動する、というフォーマット変換操作を引き起こす。

【0110】以上が、アプリケーション101が変換元ファイル130を作成して書き込み、アプリケーション102が変換元ファイル130を作成して書き込み、アプリケーション102が変換元ファイル130から変換された変換先ファイル131を読み出す場合の第2の実施例の動作である。

【0111】第2の実施例では、計算機10にある変換元ファイル130への変更が、すぐに計算機10'に伝播する。この結果、計算機10の電源がオフになったり、計算機10の二次記憶装置が故障したりして、変換元ファイル130が一時的または永続的に利用不能になっても、計算機10'の中間ファイル130'、変換先ファイル131、131'、…は計算機10'上のアプリケーションから利用可能である。

【0112】この特徴は、特に計算機10がユーザのよく利用するパーソナル・コンピュータや携帯型コンピュータで、計算機10'が多数のユーザが利用するWWWサーバなどの共有情報サーバである場合に特に効果がある。なぜなら、計算機10のユーザが計算機10の電源を切断したり、ネットワークから計算機10を切断しても、計算機10'経由で変換元ファイル130の情報を共有しているユーザにはなんら問題ないし、また、計算機10'経由で変換元ファイル130の情報を共有しているユーザがいる間でも、これらのユーザに変換先ファイル131、131'、…の利用を中断させることな

く、計算機10のユーザが変換元ファイル130を更新することができるためである。この場合、変換プログラム603は、単なるコピープログラムでも差し支えない。

【0113】変形例1：ファイル単位の変換指定

上記第1、第2の実施例では変換表120を用いて、変換元フォーマットから変換先フォーマットへの対応を行っている。しかし、本発明の実施の形態はフォーマット単位の対応づけに限定されるものではない。特に変形例1として、変換表に変換元ファイルから変換先ファイルへの対応づけを格納することができるようにしても差し支えない。これを行うには、変換表エントリ200の内部構造を、変換元フォーマット201と変換先フォーマット202と変換プログラム203の組ではなく、変換元ファイルIDと変換先ファイルIDと変換プログラム203の組とし、オープンAPIやクローズAPIで変換表120を検索する時に、ファイルIDを用いればよい。さらに進んで、実施例1の変換表120と変形例1の変換表の2つの形式を、1つの変換表に混在させる実施の方法でも差し支えない。

【0114】変形例2：変換元ファイルの隠蔽

ユーザの都合によっては、ファイルシステムが変換元ファイルを隠し、変換先ファイルのみを提供したほうが良い場面もある。例えば、第1の実施例で、変換元ファイル130が読み書き可能なフォーマットで、変換先ファイル群131、131'、…が読み出しのみ可能なフォーマットである場合、変換元ファイル130を作成したユーザは他のユーザに変換先ファイル群131、131'、…のみアクセスさせることで、変換元ファイル130を変更から守ることができる。

【0115】また第2の実施例では、計算機10'のユーザは中間ファイル130'と変換先ファイル131、131'、…とのいずれにもアクセスが可能である。しかし中間ファイル130'はファイルシステム100が使用する中間ファイルなので、中間ファイル130'を隠した方がユーザにとってわかりやすいと思われる。

【0116】このように、変換元ファイル（または特定の変換先ファイル）をユーザから隠蔽する機能を、上記第1の実施例、第2の実施例、および変形例1に追加することができる。具体的には、ファイル表122に新たな項目として「隠蔽変換元フラグ」と「隠蔽変換先リスト」を追加する。前者は変換元ファイルを隠蔽するため、また後者は変換先ファイルを隠蔽するためである。あるファイルの隠蔽変換元フラグが真なら、ファイルシステムは名前空間表121から当該ファイルのファイル名を削除する。また、隠蔽変換先リストは変換先フォーマットのリストを格納しており、上記ファイル作成APIにおいて、変換先フォーマットが、変換元ファイルの隠蔽変換先リストに含まれていれば、ステップ306において当該変換先フォーマットのファイルを名前空間表

121に追加するのを止める。

【0117】パーソナル・コンピュータ（PC）への適用例

本発明をパーソナル・コンピュータ（PC）へ適用した例を、図8を用いて説明する。なお、計算機やアプリケーションの種類や数は、例として用いたものであり、本発明の範囲を限定するものではない。

【0118】PC 800はユーザが利用する計算機である。例えばワード・プロセッサ801やWWWブラウザ802やpdf文書表示プログラム803などが、PC 800上のアプリケーションとして動作する。本発明に基づくファイルシステム100は、上述の説明通り、アプリケーションのファイル操作に伴って、必要なフォーマット変換を行う。例えば、ユーザがワード・プロセッサ801で作成した文書を保存する場合、hello.doc 804なるファイルを作成し、書き込みを行って二次記憶装置11に文書を保存する（811）。この際、ファイルシステム100は、図3のフローチャートに従って、変換先ファイルのファイル名を図1の名前空間表121（図8には図示していない）に登録する。この結果、ユーザは変換先ファイル（図8の例ではhello.html 805とhello.pdf 806）を操作できるようになる。例えば、ユーザがWWWブラウザ802を用いてhello.html 805を参照すると、WWWブラウザ802はhello.html 805を読み出す準備として、オープンAPIを発行する（821）。この際、ファイルシステム100は図4のフローチャートに従って、変換プログラム823を決定し、変換元ファイルから変換先ファイルへの変換（822）を行って変換元ファイルであるhello.doc 804から変換先ファイルであるhello.html 805の内容を得る。これによってWWWブラウザ802は、続くhello.html 805への読み出しAPI（824）によって、先ほどワード・プロセッサ801が作成したhello.doc 804の内容を、異なるフォーマットを持つファイルhello.html 805として得ることができる。なお、この一連の動作に、ユーザやアプリケーションはフォーマット変換にかかわる操作を行う必要が一切なかった点に留意すべきである。

【0119】本発明はアプリケーションによらないので、別のアプリケーション（例えばpdf文書表示プログラム803）が別の変換先ファイルの読み出しを行っても、まったく問題ない。この場合、先ほどと同様、オープンAPI（825）の発行に同期して変換プログラム827を決定し、変換元ファイルであるhello.doc 804から変換先ファイルであるhello.pdf 806へのフォーマット変換（826）を行う。これによってpdf文書表示プログラム803は、続くhello.pdf 806への読み出しAPI

（828）を行うことができる。

【0120】なお、現在PCで使用されているOSのなかには、従来のOSの機能を、複数のダイナミック・リンク・ライブラリ（DLL）の集合として実現しているOSがある。このようなOS上ではファイルシステムの主要部分もDLLとなっている。本発明をこのようなDLLベースのOSに適用する1つの方法は、ファイルシステム機能を提供するDLL（仮にFS.DLLと呼ぶ）を本発明に基づくファイルシステムの機能を提供するDLL（仮にNEW・FS.DLLと呼ぶ）で置き換える方法である。NEW・FS.DLLの中で従来のファイルシステムと同じ動作を行う部分については、FS.DLLの機能を関数呼び出しにより呼ぶことにすれば、既にある機能を再び実現することなく本発明の機能を追加することができる。すなわち、NEW・FS.DLLをFS.DLLの上にかぶせることによって、これまでFS.DLLが受けていたアプリケーションからのAPI呼び出しを、NEW・FS.DLLが受け、FS.DLLの機能が必要な場面ではNEW・FS.DLLがFS.DLLのAPIを起動する。例えば、ステップ410、ステップ504、ステップ504、ステップ504、ステップ704は、従来のファイルシステムの機能を読み出す部分なので、これらのステップでNEW・FS.DLLがFS.DLLを読み出せば良い。

【0121】WWWシステムへの適用例（1）：次に本発明をWWWシステムに適用した例を図9を用いて説明する。ネットワーク12は、既に説明したように、イントラネットやインターネットを含む計算機間の結合である。本適用例は特にTCP/IPとHTTP（Hypertext Transfer Protocol）の略）を用いるWWWに良く適合する。

【0122】この例では、本発明を組み込んだファイルシステム100を持つサーバ計算機900が、WWWサーバ911や分散ファイル・サーバ912を用いて、ネットワーク12で結合された機能限定PC 901およびPC 902にファイルを提供する。機能限定PC 901は、ネットワーク・コンピュータやThin PCと呼ばれている機能限定、低価格なパーソナル・コンピュータである。このような計算機は、低価格化を進めるために必要最低限のソフトウェアしか搭載しないのが普通である。PC 902は通常のパーソナル・コンピュータである。なお、この例では分散ファイル・サーバ912をファイルシステム100と異なるプログラムとしているが、これらを1つにした、いわゆる分散ファイル・システムを用いても差し支えない。分散ファイル・システムは分散ファイル・サーバ912がファイルシステム100に組み込まれたものと考えることができる。

【0123】ユーザは機能限定PC 901やPC 902を操作して、サーバ計算機900上のファイルにアクセスする。この例では、機能限定PC 901のユー

ザはWWWブラウザ916を用い、またPC 902のユーザはpdf文書表示プログラム917を用いてアクセスする。また、サーバ計算機900上で動作するワード・プロセッサ910を用いると、サーバ計算機900上に新たなファイルを作成、または既存のファイルを変更することができる。

【0124】ユーザがワード・プロセッサ910で作成した文書を保存する場合、ワード・プロセッサ910はファイルシステム100に変換元ファイルであるhello.doc 913の作成を依頼し(920)、書き込みを行って二次記憶装置11に文書を保存する(921)。この際、ファイルシステム100は図3のフローチャートに従って、変換先ファイルのファイル名を図1の名前空間表121(図9には図示していない)に登録する。この結果、サーバ計算機900上のアプリケーションである共有情報提供ソフトウェア(この例ではWWWサーバ911や分散ファイル・サーバ912)は変換先ファイル(図9の例ではhello.gif 914とhello.pdf 915)を操作できるようになる。

【0125】例えば、機能限定PC 901のユーザがWWWブラウザ916を用いて画像ファイルhello.gif 914を参照すると、WWWブラウザ916はhello.gif 914を読み出す要求をWWWサーバ911に送る(930)。この要求を受けたWWWサーバ911は、hello.gif 914を読み出す準備としてオープンAPIを発行する(931)。この際、ファイルシステム100は図4のフローチャートに従って変換プログラム933を決定し、変換元ファイルから変換先ファイルへの変換(932)を行ってhello.gif 914の内容をhello.doc 913から得る。これによってWWWブラウザ916は、続くhello.gif 914への読み出しAPI(934)によって、先ほどワード・プロセッサ910が作成したhello.doc 913の内容を、異なるフォーマットを持つファイルhello.gif 914として得ることができる。この結果は、機能限定PC 901上のWWWブラウザ916に返される(935)。WWWブラウザ916は画像ファイルを再生する能力を備えているので、hello.doc 913の内容を機能限定PC 901上に表示することができる。すなわち本発明によって、ユーザはhello.doc 913を作成したワード・プロセッサを搭載しない機能限定PC 901上からでも、hello.doc 913の内容を参照することができる。

【0126】同様に、別のユーザがPC 902からpdf文書表示プログラム917を用いて変換先ファイルの読み出しを行っても、まったく問題ない。たとえばpdf文書表示プログラム917が分散ファイル・サーバ912経由でhello.pdf 915を読み出す場

合、まずpdf文書表示プログラム917が分散ファイル・サーバ912にhello.pdf 915を読み出す要求を送る(940)。これを受けた分散ファイル・サーバ912は、hello.pdf 915をオープンし、読み出そうとする。この際、ファイルシステム100は上記の説明同様、オープンAPI(941)の発行に同期して変換プログラム943を決定し、hello.doc 913からhello.pdf 915へのフォーマット変換(942)を行う。これによって分散ファイル・サーバ912は、続くhello.pdf 915への読み出しAPI(944)を行い、結果をpdf文書表示プログラム917に返すことができる(945)。

【0127】この例によって、本発明が機能限定PCを含む分散計算機システム上でも有効であることを示した。

【0128】WWWシステムへの適用例(2)：さらに、WWWとPC、Thin PCを用いた分散計算機システムの別の例を、図10を用いて説明する。なお、計算機やアプリケーションの種類や数は、一例に過ぎず本発明はこれに限定されるものではない。

【0129】PC 1001を用いているユーザは、ワード・プロセッサ1010を用いて文書を作成する。この例ではhello.doc 1013を生成して(1020)書き込みを行い(1021)、書き込みが完了した時点でクローズを行う。

【0130】hello.doc 1013を生成する際、ファイルシステム100はhello.doc 1013の変換先ファイルである中間ファイルhello.doc' 1014も作成する。さらに図3のフローチャートに従って再帰的にhello.pdf 1015およびhello.gif 1016が作成される。そして最後のクローズの際に、ファイルシステム100は図7のフローチャートに従って、変換先ファイルである中間ファイルhello.doc' 1014をオープンし、クローズする。この結果、変換プログラム1023によるhello.doc 1013からhello.doc' 1014への変換が行われる(1022)。

【0131】次に機能限定PC 1003のユーザがWWWブラウザ1012を用いてhello.gif 1016をアクセスする要求をWWWサーバ1011に発行すると(1030)、この要求を受けたWWWサーバ1011はhello.gif 1016読み出しのためのオープンAPIを発行する(1031)。このオープンAPIの処理中にファイルシステム100'は、変換プログラム1033を用いたhello.doc' 1014からhello.gif 1016への変換(1032)を行う。この結果、WWWサーバ1011が続いて発行するhello.gif 1016の読み出しAPIは、hello.doc' 1014すなわちhel

hello.doc 1013から得られた最新の情報がWWWサーバ1011を経由して(1034)ユーザに渡される(1035)。

【0132】サーバ計算機1002は多くの場合多数のユーザによって共用されているので、24時間運転で運用される。PC 1001は一人のユーザの使用する計算機なので、頻繁に電源がオン、オフされるのが普通である。PC 1001の電源がオフになるなどの理由でhello.doc 1013にアクセスができない場合でも、本発明が中間ファイルhello.doc' 1014を介したフォーマット変換機能を備えている結果、機能限定PC 1003のユーザは、hello.gif 1016やhello.pdf 1015、およびhello.doc' 1014にアクセスすることが可能である。

【0133】分散検索システムへの適用例：企業内情報システムにおいて、種々の情報を検索する分散検索システムに本発明を適用した例を図11を用いて説明する。

【0134】企業内外の各所で情報提供を行うWWWサーバ計算機1101、1102、1103、…から、ユーザが所望する特定分野の情報を検索する。検索を行うのは、検索サーバ計算機1100上の検索プログラムであり、この例では全文検索サーバ1111と画像検索サーバ1112である。全文検索サーバ1111は文字列検索を行い、画像検索サーバ1112は画像のパターンマッチによる検索を行う。音声パターン検索、データベース検索など、他の検索方法も混在して構わない。WWWサーバ計算機1101、1102、1103、…および検索サーバ計算機1100はネットワーク12で結合されている。ネットワーク12は企業内のネットワーク(イントラネット)でも、企業間を繋ぐネットワークでも、さらには全世界を繋ぐインターネットのようなネットワークでも差し支えない。

【0135】企業内外のWWWサーバ計算機群からは、種々のフォーマットの情報が提供されるのが普通である。そこで検索サーバ計算機1100に本発明に基づくファイルシステム100を組み込み、各種フォーマットの違いをアプリケーションに負担をかけずに解消する。図11では、WWWサーバ計算機1101がhello.doc 1120を、WWWサーバ計算機1102がnews.pdf 1121を、WWWサーバ計算機1103がsurvey.gif 1122を、それぞれ提供している。これらはまず検索サーバ計算機1100上のWWWクライアント1110によって収集され(1123、1124、1125)、ファイルシステム100に格納される(1126、1127、1128)。なお、ファイルシステム100は二次記憶装置を持って良いが図11では省略している。

【0136】全文検索サーバ1111は文字形式ファイル「.txt」を入力として受け取るので、ファイルシ

ステム100は全文検索サーバ1111が発行するhello.txt 1133、news.txt 1136、survey.txt 1139へのファイル操作APIに応じてhello.doc 1129からhello.txt 1133を、news.pdf 1130からnews.txt 1136を、survey.gif 1131からsurvey.txt 1139を、それぞれ対応する変換プログラム(図11では図示省略)によりフォーマット変換して作成する(1132、1135、1138)。これによって全文検索サーバ1111は、本来異なるフォーマットを持つ情報に対しても検索を行うことができるようになり(1134、1137、1140)、かつ検索をかけた情報が別々のフォーマットを持っていたことを気にしなくてよい(すなわち全文検索サーバ1111内に特別なプログラミングが必要ない)。さらにユーザは、検索結果をもとに情報が最も豊かな変換元ファイル(hello.doc 1129、news.pdf 1130、survey.gif 1131)の情報にアクセスすることができる。この場合にもファイルシステム100は、変換元ファイルをユーザが使うアプリケーションが利用可能なフォーマットに変換することにより、ユーザの利便を図ることができる。

【0137】さらに、同じ情報源を画像に変換することにより、まったく異なる検索方法を実施することが可能である。画像検索サーバ1112は画像形式ファイル「.gif」を入力として受け取るので、ファイルシステム100は画像検索サーバ1112が発行するhello.gif 1142、news.gif 1145、survey.gif 1131へのファイル操作APIに応じてhello.doc 1129からhello.gif 1142を、news.pdf 1130からnews.gif 1145を、それぞれ対応する変換プログラム(図11では図示省略)によりフォーマット変換して作成する(1141、1144)。これによって画像検索サーバ1112は、本来異なるフォーマットを持つ変換元ファイル(1129、1130、1131)の情報に対して検索を行うことができ(1143、1146、1147)、かつ検索をかけた情報が別々のフォーマットを持っていたことを気にしなくてよい。

【0138】電子商取引システムへの適用例：企業と企業、または企業と個人の間の取り引きを電子的に行う電子商取引(EC)システムに本発明を適用した例を図12を用いて説明する。

【0139】例えばインターネット経由でECを行う場合には、請求書はいくつもの企業や団体のネットワークを経由して第1のユーザから第2のユーザに届けられる。この際、第1、第2のユーザのプライバシーを保護する目的で、また請求書が第三者によって不当に改竄さ

れるのを防ぐ目的で、請求書を暗号化して送ることがよく行われる。

【0140】ECサーバ計算機1200はECクライアント計算機1201と通信して電子商取引を行う。ECサーバ計算機1200を使用する第1のユーザが、ECクライアント計算機1201を使用する第2のユーザの注文に対する請求書を作成し、第2のユーザがそれを受け取る場面を説明する。このような場面での暗号化（および復号化）は、一種のフォーマット変換と考えることができる。この場合の変換プログラムは、暗号化プログラムや復号化プログラムである。

【0141】第1のユーザは過去に第2のユーザから受け取った注文に対する請求書であるinvoice.doc 1220をワードプロセッサ1211で作成する（1230）。そしてinvoice.doc 1220を暗号化プログラム1212で暗号化し（1231）、暗号化後のファイルであるinvoice.doc.crp 1221を電子メール（または電子メールの添付ファイル）で第2のユーザに送る（1233）。これを受け取った第2のユーザは、電子メールをファイルシステムにinvoice.doc.crp 1221'として保存する。この際、本発明のファイルシステムが、復号化プログラム1213を変換プログラムとし、invoice.doc.crp 1221'を変換元ファイルとしたフォーマット変換操作を行い（1234）、変換先ファイルとしてinvoice.doc 1220'を第2のユーザに提供する（1235）。これにより第2のユーザは、invoice.doc 1220'を手動で復号化する手間を負うことなく、ワードプロセッサ1211'を用いて請求書を参照することができる（1236）。またこの例では示さなかったが、第1のユーザが請求書の暗号化を行う場面でも本発明を適用することができる。これにより、第1のユーザ、第2のユーザとも請求書の暗号化と復号化を気にする必要なく、高速かつ快適に電子商取引を行うことができる。

【0142】

【発明の効果】ユーザはアプリケーション上で行うユーザの本来の作業を行えばよくなり、その過程で必要となる様々なフォーマット変換（1段階でも、多段階でも良

い）はユーザの関与なしで行われる。この際、変換元ファイルを指定したり、フォーマット変換のタイミングを指定する必要がなく、ユーザは常に最新の変換先ファイルを利用可能となる。さらに、変換元ファイルと変換先ファイル群との間の一貫性が保持される。また変換先ファイルを多く保持することによる二次記憶領域の無駄を解消できる。万一変換元ファイルが利用不能な間もフォーマット変換が行える。

【図面の簡単な説明】

【図1】第1の実施例の内部構成の概略を示すブロック図。

【図2】第1の実施例で用いるデータ構造を示す図。

【図3】ファイル作成APIの動作を説明するフローチャート。

【図4】オープンAPIの動作を説明するフローチャート。

【図5】クローズAPIの動作を説明するフローチャート。

【図6】第2の実施例の内部構成の概略を示すブロック図。

【図7】第2の実施例で変換元ファイルのクローズAPIの動作を説明するフローチャート。

【図8】本発明のパーソナル・コンピュータへの適用例を示す図。

【図9】本発明のWWWシステムへの適用例を示す図。

【図10】本発明のWWWシステムへの別の適用例を示す図。

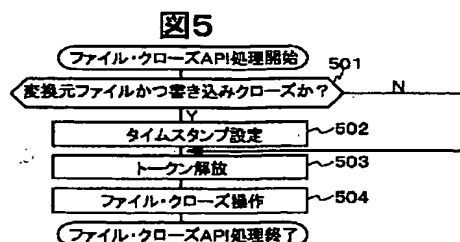
【図11】本発明の分散検索システムへの別の適用例を示す図。

【図12】本発明の電子商取引システムへの適用例を示す図。

【符号の説明】

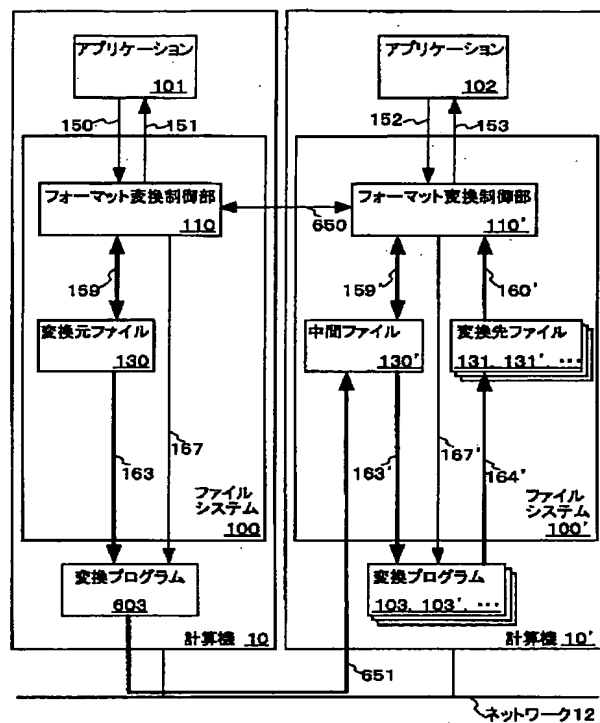
10：計算機、11：二次記憶装置、100：ファイルシステム、101、102：アプリケーション、103、103'、…：変換プログラム、104：設定プログラム、110：フォーマット変換制御部、111：フォーマット変換設定部、120：変換表、121：名前空間表、122：ファイル表、123：トークン表、124：削除候補表、130：変換元ファイル、131、131'、…：変換先ファイル。

【図5】



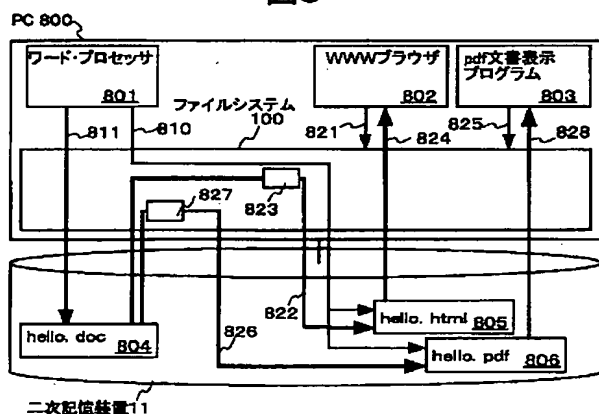
【図6】

図6



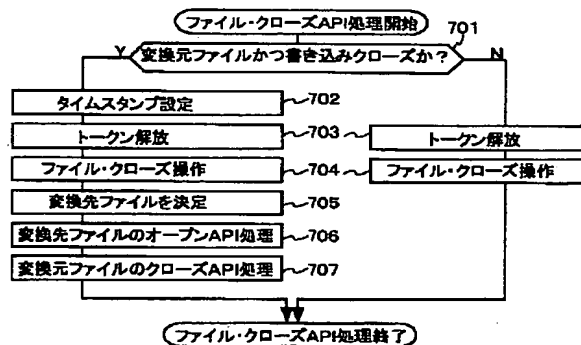
【図8】

図8



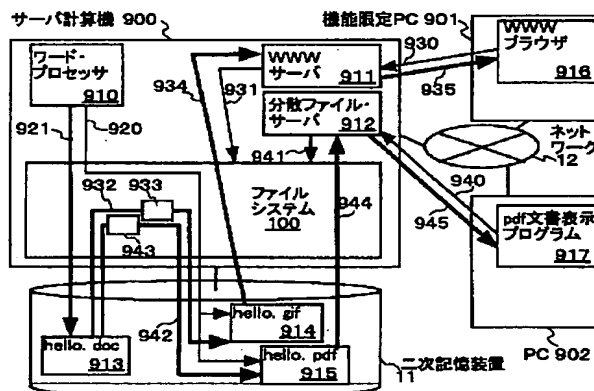
【図7】

図7



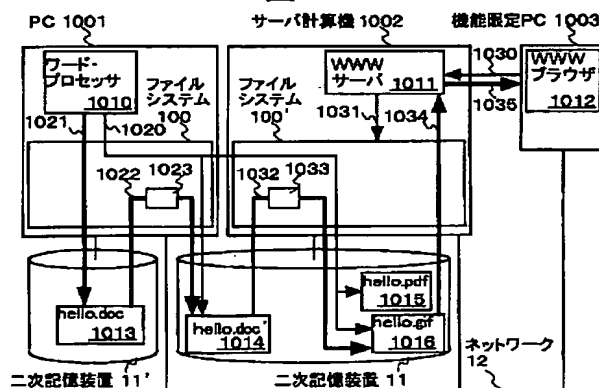
【図9】

図9



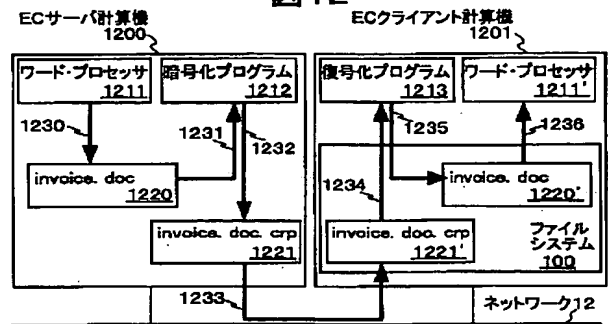
【図10】

図10



【图 1 2】

图12



(72)発明者 関 京華
東京都国分寺市東恋ヶ窪一丁目280番地
株式会社日立製作所中央研究所内

**This Page is Inserted by IFW Indexing and Scanning
Operations and is not part of the Official Record**

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☒ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.

As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.